Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 21 (2013) 351 – 359

Procedia Computer Science

The 3rd International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH-2013)

The Future of Mobile E-health Application Development: Exploring HTML5 for Context-aware Diabetes Monitoring

Davy Preuveneers[a], Yolande Berbers[a], Wouter Joosen[a]

[a]*iMinds-DistriNet, KU Leuven, 3001 Leuven, Belgium*

Abstract

According to predictions of information technology research and advisory firms, such as Gartner, hybrid HTML5 applications will be the future for mobile application development. In this paper, we explore the feasibility of using HTML5 and related web application standards for the development of mobile e-health applications, using a diabetes monitoring application as a practical use case. Context-awareness and visualizing multivariate data with parallel coordinates for decision support are key features of this mobile e-health application. We compare the strengths and weaknesses of the hybrid HTML5 approach with native mobile applications, and report on practical experiences with the development and usage of the application. Our experiments show that developers of fairly advanced context-aware mobile applications can definitely benefit from the HTML5 application portability across different mobile platforms. However, compared to native applications, they should be aware of missing features such as secure storage, non-negligible performance penalties of JavaScript business logic on a mobile platform, and an inferior user experience.

Selection and peer-review under responsibility of Elhadi M. Shakshuki

Keywords:
e-health, context-awareness, mobile applications, patient monitoring, data visualization, html5

1. Introduction

With a growing population of patients, the pressure on healthcare and welfare systems will continue to increase. Many research initiatives and multidisciplinary research fields that leverage both ICT and healthcare disciplines are therefore looking at recent advances on the information and communication technologies front to transform the way we go about personalized healthcare and independent living. Significant strides in wireless medical and environmental sensors promise to deliver patients and healthcare professionals with novel cost-effective solutions to health management anytime and anywhere, truly enabling the vision of mobile and pervasive healthcare. In this paper, we focus on context-aware applications for patient monitoring and care, data visualization and decision support. Related to these challenges, we can witness two different trends in e-health application development.

First, the ubiquitous computing community has spent significant efforts over the past decade on researching data fusion and artificial intelligence techniques to create a variety of context-aware applications and smart environments. These components are now finding their way into e-health and personal health monitoring systems, albeit still often with a *one sensor for one disease* approach. The key challenge is to

1877-0509 © 2013 The Authors. Published by Elsevier B.V.
Selection and peer-review under responsibility of Elhadi M. Shakshuki
doi:10.1016/j.procs.2013.09.046

raise *context awareness* to a reusable higher-level layer of abstraction such that personal health monitoring systems can more easily leverage individual and situational information (current time, location, activity, etc.) to really empower patients with the self-management of their health.

Second, despite the many advances in sensor technology and telemonitoring, activity recognition and human computer interaction, convincing solutions are still missing. This observation is not only triggered by the inherent development cost and complexity of engineering such systems, but also because of the way the assistive capabilities are sometimes ill perceived when patients do not comprehend why an application offers a certain suggestion. Good *data visualization* techniques for *decision support* are key to give the individual a feeling of still being in control.

With smartphones and tablets likely to overtake the PC market in the next couple of years and HTML5 being an enabling technology that promises to work seamlessly across mobile platforms and browsers, Gartner claims in its IT predictions for 2013 report [1] that the future for web application consumption is mobile. In this paper, we report on practical experiences with using HTML5 and related web technologies to deliver context-aware personal health assistance applications, and the challenges with targeting such smart e-health applications to mobile devices.

After reviewing related work in section 2, we present our context-aware diabetes monitoring assistant in section 3 as a motivating example for a mobile e-health application. Section 4 discusses various implementation details using HTML5 web application technologies. In section 5 we evaluate the feasibility of the approach and highlight some practical experiences with the development and the usage of the application on mobile platforms. We conclude in section 6 summarizing the main insights and identifying possible topics for future work.

2. Related Work

In recent years, mobile devices have received a lot of attention as a platform for pervasive computing research. Providing a detailed review of the state-of-the-art on context-awareness and e-health web applications is beyond the scope of this paper. We refer the interested reader to a survey [2] by Bricon-Souf and Newman on how context-awareness is being applied in various healthcare projects. Instead, we review related work in the fields of feasibility analysis of HTML5-based applications and relevant performance and security concerns.

The feasibility of HTML5 for mobile devices was explored by Juntunen et al. in [3], focusing on features like graphics, multimedia, user interaction, device adaptation, network, battery, and other performance aspects. They conclude that HTML5 applications do not achieve the same performance of native applications, but that the lower cost and cross-platform availability of web applications are clear advantages. Xinogalos et al. [4] specifically focus on advantages and disadvantages of using HTML5, exploring similar features in a mobile cloud computing setting.

De Ryck et al. scrutinized in [5] new HTML5 standards for potential security problems and carried out a systematic analysis on how new web mechanisms might break the security of existing web applications, and how gracefully different newly proposed mechanisms interact with each other. They found various violations against these security goals and related issues in major browsers. In [6], West et al. specifically focus on the Web Storage specification and discuss privacy, security, and performance concerns of the localStorage and sessionStorage attributes.

In this paper, we investigated similar issues with the applicability of HTML5 and JavaScript technologies for an advanced mobile and context-aware health self-care application, and carried out a comparative performance analysis for certain activity recognition, visualization and security features.

3. Motivating scenario: context-aware patient monitoring for diabetes self-management

In prior work [7], we presented a case study on using a smartphone as a mobile health management companion for people diagnosed with type 2 diabetes. The specific aim of this application was to not just replace a paper log book with a portable electronic device, but to capture and integrate the relevant user

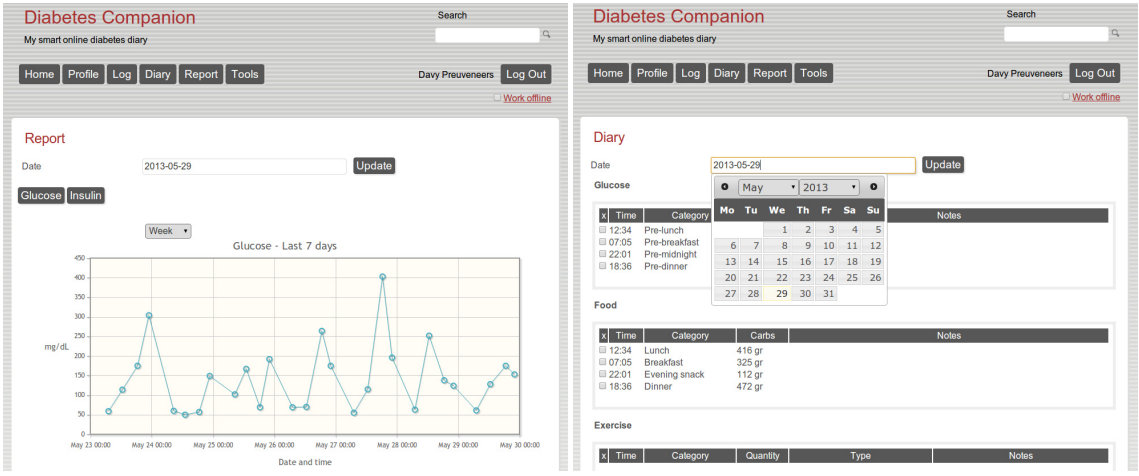


Fig. 1. The hybrid HTML5 implementation of our Diabetes Companion application

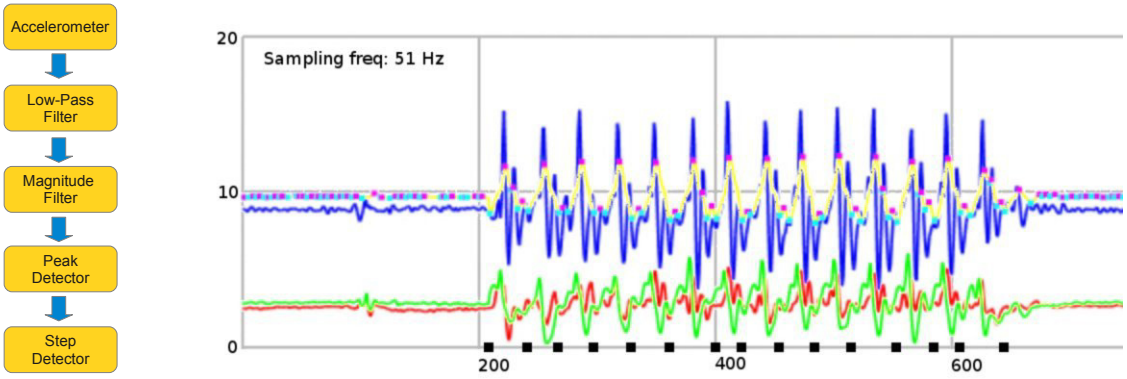


Fig. 2. Accelerometer-based based motion activity recognition and step counting

context into the data log in order to better compare measurements with previous situations. The augmented data logs and similarity analysis helped find trends and advise the individual more accurately and tailored to the current situation. The major challenge was to identify classes of activities, such as eating and exercise, that have an effect on blood glucose levels. As *activities of daily living* (ADL) typically present recurring behavioral patterns, we investigated the presence of correlations between *time* and *location* on the one hand and types of activities on the other hand, and used these correlations to find similar situations of the past as a recommendation for the patient. The advantage of this approach is that the system offers personalized assistance, not only leveraging the results of the blood glucose sensor itself, but also basing its advice on the current context and anticipating what is likely going to happen next. A revised and improved Android application was recently developed in the frame of the FP7 BUTLER[1] project. The following sections discuss challenges with implementing similar functionality in HTML5, CSS3 and JavaScript web standards (see Figure 1).

[1] `http://www.iot-butler.eu`

4. Developing smart and mobile e-health applications with HTML5

4.1. Context-awareness: time, location and activity recognition

Context-awareness is defined as the capability of being able to perceive the current situation and to respond seamlessly and proactively. Time and location are usually first class entities in context management. Beyond time and location we use the tri-axial accelerometer readily embedded in many smartphones as a pervasive, non-intrusive and mobile sensing unit for activity recognition. However, compared to time and location which are easy to sense directly on a mobile platform, we need fairly more complex and computationally intensive algorithms to classify the activity of the user (standing, walking, running, falling, etc.) and track the number of steps taken each day as a measure for the exercise level and active lifestyle of the user. Figure 2 provides on the left an overview of the composition of the activity recognition components and the dataflow among them. On the right, the figure visualizes the X,Y,Z accelerometer data and intermediate outputs after processing by the components:

- **Accelerometer**: This component produces a continuous stream of acceleration data, consisting of X,Y and Z values arriving at a certain sampling rate *(cfr. red, green and blue lines)*.
- **Low-pass filter**: For mobility tracking, e.g. walking or running, we are mainly interested in acceleration peaks that arrive at a frequency of maximum 5Hz (i.e. max 5 steps per second). For this purpose, we use a 'moving average' component as a simple low-pass filter to remove high-frequency noise.
- **Magnitude filter**: In many cases, we do not know how the accelerometer is oriented at the offset of the motion activity. Furthermore, the orientation of the sensor is subject to change while moving around. Therefore, we carry out the signal analysis on the overall magnitude of the acceleration signal *(cfr. yellow line)*.
- **Peak filter**: A single step is characterized by a pattern of several maxima and minima in the time domain of the acceleration signal. This component extracts these features in the signal for further analysis *(cfr. cyan and magenta markers)*.
- **Step detector**: This component identifies the same peak for every step in order to correctly count the number of steps and to differentiate between standing still, walking and running (i.e. the peak rate) *(cfr. black markers)*.

Next to the step counting components, our context middleware has additional components for fall detection:

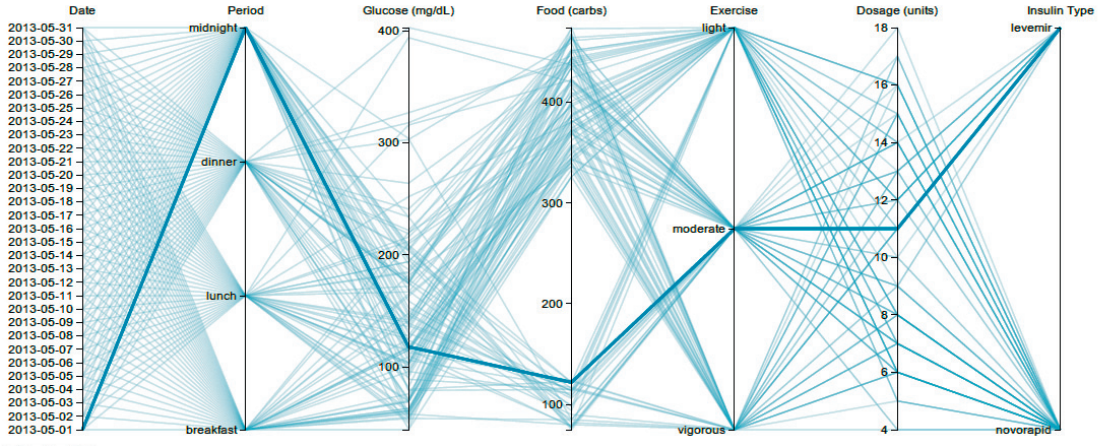
- **High-pass filter**: Falls are characterized by sudden and high-frequency changes of the acceleration signal, both in amplitude and orientation. This component implements a high-pass Finite Impulse Response (FIR) filter to extract these features.
- **Signal Magnitude Area filter**: This component analyzes the signal magnitude area (SMA) of the high-frequency part of the acceleration signal, and identifies a fall if this feature passes a threshold.

The above components are merely a subset of the functionality that is part of our context middleware. The objective is not to discuss these components in detail, but to highlight the challenges of integrating these algorithms and functionality in an HTML5 web application environment.

4.2. A hybrid HTML5 approach for mobile context-aware applications

Time and location are contextual concepts that are easily accessible in standard HTML5 web applications. For example, the W3C Geolocation API Specification[2] defines a high-level interface to location information of the device hosting the application. The position information is offered in terms of World Geodetic System coordinates and the API is agnostic of the underlying location information sources (GPS,

[2] `http://www.w3.org/TR/geolocation-API`



Showing all 124 rows

Date	Time	Period	Glucose (mg/d...	Food (carbs)	Exercise	Dosage (units)	Insulin Type	id
2013-05-01	07:04	breakfast	69	350	vigorous	7	novorapid	0
2013-05-01	12:36	lunch	104	423	moderate	6	novorapid	1
2013-05-01	18:29	dinner	71	474	vigorous	13	novorapid	2
2013-05-01	**22:26**	**midnight**	**118**	**122**	**moderate**	**11**	**levemir**	**3**
2013-05-02	07:07	breakfast	54	341	light	6	novorapid	4
2013-05-02	12:38	lunch	55	414	moderate	4	novorapid	5
2013-05-02	18:15	dinner	175	473	light	15	novorapid	6
2013-05-02	22:48	midnight	159	99	moderate	12	levemir	7
2013-05-03	08:31	breakfast	63	354	vigorous	7	novorapid	8
2013-05-03	13:03	lunch	194	421	moderate	7	novorapid	9
2013-05-03	18:27	dinner	82	444	moderate	14	novorapid	10

Fig. 3. Data visualization with parallel coordinates

WiFi, GSM/CDMA cell IDs, etc.). However, access to other sensors like the accelerometer requires a hybrid HTML5 approach. Hybrid applications combine the portability of HTML5 web applications with a native container (e.g. Android or iOS) to better leverage native mobile device capabilities. In the same report referred to earlier [1], Gartner estimates that by 2016 more than 50 percent of mobile apps deployed will be hybrid. They confirm that the need for context awareness in mobile applications has increased with the capabilities of mobile devices, causing developers to consider both hybrid and native architectures.
To build the hybrid HTML5 variant of our diabetes e-health application, we make use of the PhoneGap 2.7.0 [3] cross-platform development tool to target Android and iOS devices using the same HTML5 and CSS code for rendering and JavaScript code for the business logic. To simplify the development of the user interface and to support better cross-platform interoperability, we also make use of the jQuery UI framework [4] and related plugins.

4.3. Intelligibility for decision support through data visualization with parallel coordinates

Thinking is a sign of being smart, but thinking that you are smart does not make it so. Nothing is more frustrating than a smart application that takes automated actions or gives recommendations that is so confident of its own righteousness that it becomes intolerant of concerns or feedback from its users. If an application behaves in a way that does not seem logical to the patient, it usually reflects an imbalance between what the application knows and what the person knows. Being able to understand why an application acts in a certain way is a first step towards being able to correct undesired behavior. Any smart application or system that puts the human in the loop requires a clear communication of its internal reasoning logic. This

[3] http://phonegap.com/
[4] http://jqueryui.com/

means it must provide causal explanations of which contextual information has been acquired and which rules governed its behavior, preferably at a degree of detail that the user understands. Furthermore, it must offer users with appropriate mechanisms to incorporate their feedback to modify the system's behavior. This concept is referred to as *intelligibility* and was introduced in 2009 by Dey and Newberger [8] with follow-up work for uncertain context-aware applications by Lim et al. [9].

Visualizing data is important to implement intelligibility in our application. A diabetes patient chooses his insulin dosage based on a variety of parameters, including a default patient specific insulin dosage plan, the current blood glucose level, the amount of carbohydrates in his next meal (derived from nutrition tables) and the intensity of physical activities (counting the pace and the amount of steps), etc. Our application keeps track of this data to filter for similar situations and scenarios of the past as a suggestion for the patient. Visualizing the similarity as an explanation of why the suggestions are relevant is important to increase the confidence in the application. We used *parallel coordinates* [10] as a common way of visualizing and analyzing high-dimensional multivariate data, and relied on the JavaScript implementation[5] by Kai Chang et al. on top of the D3.js (Data-Driven Documents) framework[6]. Figure 3 illustrates the concept of parallel coordinates for our diabetes application data. The date and period of the day, the glucose value, the number of carbs of the meal, the intensity of the exercise level, the insulin dosage, etc. form the different dimensions. Rather than representing each entry as a point in an n-dimensional space, each entry is now depicted as a polyline with vertices on the parallel axes (representing the different dimensions). The position of the vertex on the i-th axis corresponds to the i-th coordinate of the data point.

The current data visualization implementation provides different user interaction capabilities. The user can visually select a range on each of the axes. In that case, rather than showing all entries, only those entries will be visualized and represented in the table below whose polylines pass through the selected ranges. Furthermore, the user can also select an entry in the table, and the corresponding polyline will be highlighted in the graph accordingly. This data visualization and visual querying mechanism are very powerful decision support tools, but require a reasonably sized display from a usability point of view.

5. Feasibility analysis and comparative evaluation

In this section, we will evaluate the feasibility of using HTML5 and related web application standards for the development of context-aware and mobile e-health applications and compare the strengths and weaknesses of the hybrid HTML5 approach with native mobile applications. We will also report on practical experiences with the development and usage of the application. We will consider the case when the HTML5 web application runs within a mobile web browser, and as a hybrid application in a native container. All mobile performance experiments are carried out on an ASUS Transformer TF101 Android tablet with a Dual-core 1 GHz Cortex-A9 CPU and an Nvidia Tegra 2 chipset. The tablet runs Ice Cream Sandwich v4.0 as operating system and the Google Chrome v26.0 and Firefox v21.0 web browsers for Android. For comparative benchmarking, the application was also tested running in the Firefox and Google Chrome browsers on a desktop with a Intel Core2 Duo 3.16GHz CPU.

5.1. Offline use and secure local storage for structured data

The diabetes monitoring application needs local storage capabilities to boost performance or to permanently save historic data for later analysis. Persistent local storage is one of the areas where native applications have an edge over web applications. Recent web browsers support the HTML5 Web Storage[7] specification for storing key-value pairs on the client side in a way that is fairly similar to HTTP session cookies. However, major weaknesses with this persistency model are:

- The key-value pair model is not well-suited for storing and querying structured data. Additional layers are necessary to implement database functionality, i.e. select records, add records, delete records, drop tables, etc.

[5] http://syntagmatic.github.io/parallel-coordinates/
[6] http://d3js.org/
[7] http://dev.w3.org/html5/webstorage/

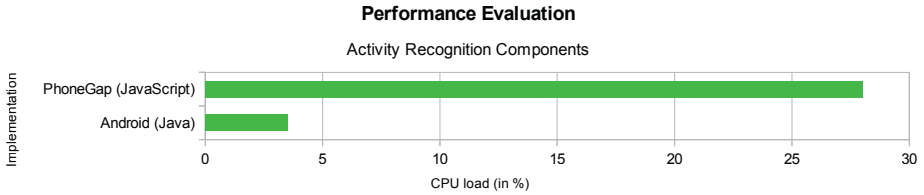


Fig. 4. Performance comparison of the Java and JavaScript implementations of the activity recognition components

- Although the limit is not standardized, most browsers only offer 5 megabytes of storage space per origin. This also means that web applications with the same origin (scheme, host and port of a url) share their local storage.
- There are no standardized APIs to ascertain the amount of storage space currently being in use or still available. An exception QUOTA_EXCEEDED_ERR is thrown if an application tries to exceed the storage quota.
- Data is stored unencrypted, which is a major security concern if one is dealing with sensitive data. If there is a XSS vulnerability[8] in the application, anything stored in local storage is available to an attacker.

At the moment there are no viable alternatives for persistent storage. The Web SQL Database[9] specification is no longer maintained due to lack of multiple independent implementations, and the Indexed Database API[10] specification is not yet widely supported on mobile platforms (i.e. not supported on iOS Safari and the default Android browser, but supported in Chrome and Firefox). For our hybrid HTML5 application, PhoneGap offers storage capabilities following the W3C Web SQL Database and W3C Web Storage specifications, implementing them for those mobile platforms that lack support for it natively.

In our current mobile diabetes monitoring application, data persistency is built on top of the W3C Web Storage specification, with an additional lightweight layer in JavaScript that mimics minimal database functionality. Data is stored securely using AES encryption with a key derived from the user's password using the PBKDF2 password strengthener. The Stanford Javascript Crypto Library[11] provides implementations of these cryptographic algorithms.

5.2. Performance of JavaScript-based context management and encryption

We compared the performance of the different versions of our activity recognition components responsible for processing the accelerometer data. The Java implementations of these algorithms run directly within the Dalvik virtual machine on Android, whereas the JavaScript implementations run within the PhoneGap wrapper for Android (i.e. they are executed by the JavaScript engine of the Android WebView embedded browser component). We tested the PhoneGap JavaScript and the Android Java implementations of the activity recognition components in a 15 minute trial run. The averaged performance results in Figure 4 show that the CPU load for the JavaScript implementation is significantly higher (28% vs. 3.5%). Accelerometer-based activity recognition is a continuous process, and while the JavaScript implementation performed accurately, the higher CPU load caused a quicker battery drain. For simple context types that require little processing, like time and location, the performance difference was negligible.

Another experiment was carried out to measure the performance overhead of the additional AES encryption layer, transparently overriding the getItem() and setItem() APIs of the local W3C Web Storage framework with encryption enabled ones. This way, the user's profile is stored securely as encrypted key-value

[8] https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)
[9] http://dev.w3.org/html5/webdatabase/
[10] http://www.w3.org/TR/IndexedDB/
[11] http://crypto.stanford.edu/sjcl/

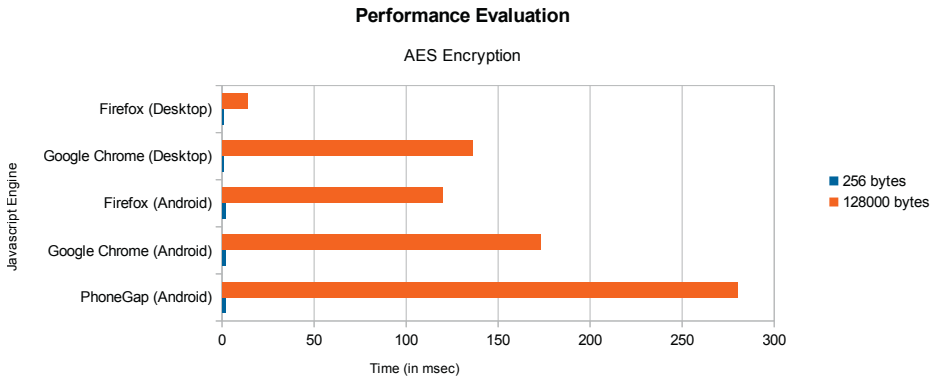


Fig. 5. Performance comparison of the JavaScript-based AES implementation

pairs. As the database containing the health data is also stored as an item using a string-based JSON-like structure, each and every update requires the full encryption of the database. We carried out 10 runs of the performance experiment each with 2 different lengths of item values, and this on a variety of browsers. The performance results are shown in Figure 5. The encryption time for short key-value pairs (e.g. less than 256 bytes) is not noticeable (in the order of a millisecond or less). For larger key-value pairs (e.g. a JSON database representation of about 128000 bytes), the duration is acceptable as long as the updates do not appear frequently.

5.3. User experience and visualization

As data visualization is an important component in our application, we set up a preliminary user study where 9 participants had the opportunity to experiment with the HTML5 application. The visualization and interaction capabilities were tested on the desktop with the Firefox and Google Chrome browsers. On the Android tablet, the application was deployed as a standalone PhoneGap application as well as tested with the Firefox and Google Chrome browsers for Android. Next to the aforementioned mobile devices, the following smartphones were also used in this assessment:

- Samsung Galaxy S3
- Samsung Galaxy S4
- Google Nexus 4

Although the PhoneGap API allows for specifying any delay value for accelerometer updates, experiments with the above devices have shown that the actual refresh rate did not quite match the requested sampling frequency (up to relative errors of 50%), jeopardizing especially the accuracy of our activity recognition components relying on frequency domain features (i.e. features obtained after a Fourier transformation).

The interactive parallel coordinates-based data visualization of Figure 3 was perceived as very powerful, though more suited for devices with large displays (tablets and desktops/laptops) and with a steeper learning curve compared to other single dimensional visualizations as in Figure 1. Whereas the overall user experience on the desktop was considered positive, a noticeable user interface lag was reported on the tablet, especially when interacting with the interactive parallel coordinates chart (see Figure 3). Furthermore, on some devices the PhoneGap variant of our application – which leverages the default browser component of the device – had trouble rendering our graphs (see left part of Figure 1).

Although no quantitative results are available, the mobile browsers – i.e. Firefox and Google Chrome – were reported to provide a slightly smoother user interaction experience compared to the PhoneGap application which relies on the default embedded browser component of Android. However, further experiments with other devices are necessary to properly identify whether any bottlenecks while interacting with the parallel coordinates charts are due to performance concerns with the JavaScript engine or rendering issues with the browser component.

6. Conclusion

In this paper, we reported on practical experiences with using HTML5 and related web technologies to deliver context-aware personal health assistance, using a mobile diabetes application as a motivating use case. The fact that our context-aware diabetes application runs unmodified in a variety of mobile and desktop browsers, as well as packaged as a standalone mobile application, clearly shows the reduced development effort and the cross-platform capabilities as major benefits of HTML5. Furthermore, our experiments show that implementing context management in JavaScript is feasible as long as no computationally intensive operations need to be carried out continuously, as our accelerometer-based activity recognition algorithms in JavaScript turned out to be more than an order of magnitude more expensive compared to equivalent native implementations in Android. Other experiments involving encryption performance and user experience testing showed that there is still a significant performance gap when executing HTML5 web applications on mobile devices versus in browsers in regular desktops.

Compared to native applications, developers should be aware of feature inconsistencies across browsers, limitations such as the 5MB storage space per origin, subpar database functionality, or missing security features such as encryption. While it is feasible to implement missing native logic with additional JavaScript code, such an approach comes with a performance cost. However, once the HTML5 specification is fully finalized, we expect that web browsers and frameworks for hybrid HTML5 applications will address most of these concerns.

To mitigate performance concerns with computationally expensive code, we are investigating distributed life-cycle support for JavaScript-based applications to dynamically offload complex tasks from a mobile to a resource rich environment such as the cloud. We are building upon the Java Scripting API of Java SE to run part of the JavaScript business logic of the mobile application server-side if connected and low on resources. However, the challenge is take into consideration any communication and computational trade-offs when transmitting data for remote processing.

Acknowledgments

This research is partially funded by the Research Fund KU Leuven, and by the EU FP7 projects BUTLER and NESSoS. With the financial support from the Prevention of and Fight against Crime Programme of the European Union (B-CCENTRE).

References

[1] K. Dulaney, V. L. Baker, M. Hung, D. M. Smith, Predicts 2013: Mobility becomes a broad-based ingredient for change, Tech. rep., Gartner (November 2012).
URL http://www.gartner.com/id=2242915

[2] N. Bricon-Souf, C. R. Newman, Context awareness in health care: A review, I. J. Medical Informatics 76 (1) (2007) 2–12.

[3] A. Juntunen, E. Jalonen, S. Luukkainen, Html 5 in mobile devices – drivers and restraints, 2013 46th Hawaii International Conference on System Sciences 0 (2013) 1053–1062. doi:http://doi.ieeecomputersociety.org/10.1109/HICSS.2013.253.

[4] S. Xinogalos, K. E. Psannis, A. Sifaleras, Recent advances delivered by html 5 in mobile cloud computing applications: a survey, in: Proceedings of the Fifth Balkan Conference in Informatics, BCI '12, ACM, New York, NY, USA, 2012, pp. 199–204. doi:10.1145/2371316.2371355.
URL http://doi.acm.org/10.1145/2371316.2371355

[5] P. D. Ryck, L. Desmet, F. Piessens, W. Joosen, A security analysis of emerging web standards - html5 and friends, from specification to implementation, in: P. Samarati, W. Lou, J. Zhou (Eds.), SECRYPT, SciTePress, 2012, pp. 257–262.

[6] W. West, S. M. Pulimood, Analysis of privacy and security in html5 web storage, J. Comput. Sci. Coll. 27 (3) (2012) 80–87.
URL http://dl.acm.org/citation.cfm?id=2038772.2038791

[7] D. Preuveneers, Y. Berbers, Mobile phones assisting with health self-care: a diabetes case study, in: G. H. ter Hofte, I. Mulder, B. E. R. de Ruyter (Eds.), Mobile HCI, ACM International Conference Proceeding Series, ACM, 2008, pp. 177–186.

[8] A. K. Dey, A. Newberger, Support for context-aware intelligibility and control, in: D. R. O. Jr., R. B. Arthur, K. Hinckley, M. R. Morris, S. E. Hudson, S. Greenberg (Eds.), CHI, ACM, 2009, pp. 859–868.

[9] B. Y. Lim, A. K. Dey, Investigating intelligibility for uncertain context-aware applications, in: J. A. Landay, Y. Shi, D. J. Patterson, Y. Rogers, X. Xie (Eds.), Ubicomp, ACM, 2011, pp. 415–424.

[10] A. Inselberg, Parallel Coordinates: Visual Multidimensional Geometry and Its Applications, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2009.